Sum-to-Modified Booth (S-MB) Recoding Schemes using 4:2 compressors

Naveen Kumar N M, Associate Prof. Aby Thomas

naveenmenon666@gmail.com, +91 9567567737

Abstract— Complex arithmetic operations are widely used in Digital Signal Processing (DSP) applications. In this work, we focus on optimizing the design of the fused Add-Multiply (FAM) operator for increasing performance. We investigate techniques to implement the direct recoding of the sum of two numbers in its Modified Booth (MB) form using 4:2 compressors**.** Introduction of a structured and efficient recoding technique and exploring three different schemes by incorporating them in FAM designs. Comparing to the existing recoding schemes, the proposed technique yields considerable reductions in terms of critical delay, hardware complexity and power consumption of the FAM unit**.**

Keywords— Booth Algorithm – Modified Booth Encoder – S-MB recoder – Sign Extension Algorithm – Fused Add-Multiply unit – Signed Half Adder and Full Adder - Compressors.

INTRODUCTION

Modern consumer electronics make extensive use of Digital Signal Processing (DSP) providing custom accelerators for the domains of multimedia, communications etc. Typical DSP applications carry out a large number of arithmetic operations as their implementation is based on computationally intensive kernels, such as Fast Fourier Transform (FFT), Finite Impulse Response (FIR) filters etc. As expected, the performance of DSP systems is inherently affected by decisions on their design regarding the allocation and the architecture of arithmetic units. Recent research activities in the field of arithmetic optimization have shown that the design of arithmetic components combining operations which share data, can lead to significant performance improvements. Based on the observation than the addition can often be subsequent to a multiplication, the Multiply Accumulator (MAC) and Multiply-Add (MAD) units were introduced leading to more efficient implementations of DSP algorithms compared to the conventional ones, which use only primitive resources.

Several architectures have been proposed to optimize the performance of the MAC operation. As noted, MAC components increase the flexibility of DSP data path Synthesis as a large set of arithmetic operations can be efficiently mapped onto them. Except the MAC/MAD operations, many DSP applications are based on Add-Multiply (AM) operations. The straightforward design of the AM unit, by first allocating an adder and then driving its output to the input of a multiplier, increases significantly both area and critical path delay of the circuit. Targeting an optimized design of AM operators, fusion techniques are employed based on the direct recoding of the sum of two numbers (equivalently a number in carry-save representation) in its Modified Booth (MB) form. Thus, the carry-propagate (or carry-look-ahead) adder of the conventional AM design is eliminated resulting in considerable gains of performance.

Another Research paper presented a signed-bit MB recoder which transforms redundant binary inputs to their MB recoding form. A special expansion of the preprocessing step of the recoder is needed in order to handle operands in Carry-save representation. In a paper, the author proposes a two-stage recoder which converts a number in carry-save form to its MB representation. The first stage transforms the carry-save form of the input number into signed-digit form which is then recoded in the second stage so that it matches the form that the MB digits Request. Recently, this technique has been used for the design of high performance Flexible coprocessor architectures targeting the computationally intensive DSP Applications. A present optimized design of previous paper which results in Improvements in both area and critical path.

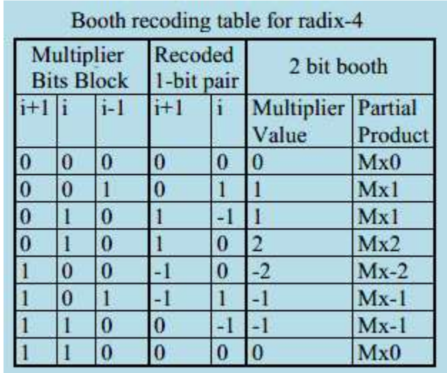
More specifically, we propose a new recoding technique which decreases the critical path delay and reduces area and power consumption. The proposed *S-MB* algorithm is structured, simple and can be easily modified using compressor circuits in order to be applied either in signed (in 2’s complement representation) or unsigned numbers, which comprise of odd or even number of bits. We explore three alternative schemes of the proposed *S-MB* approach using conventional and signed-bit Full Adders (FAs) and Half Adders (HAs) as building blocks. We evaluated the performance of the proposed *S-MB* technique by comparing its three different schemes with the state-of the- art recoding techniques.

BOOTH ALGORITHM

Booth multiplication algorithm or Booth algorithm was named after the inventor Andrew Donald Booth. It can be defined as an algorithm or method of multiplying binary numbers in two’s complement notation. It is a simple method to multiply binary numbers in which multiplication is performed with repeated addition operations by following the booth algorithm. Again this booth algorithm for multiplication operation is further modified and hence, named as modified booth algorithm.

MODIFIED BOOTH ENCODER

This modified booth multiplier is used to perform high-speed multiplications using modified booth algorithm. This modified booth multiplier's computation time and the logarithm of the word length of operands are proportional to each other. We can reduce half the number of partial product. Radix-4 booth algorithm used here increases the speed of multiplier and reduces the area of multiplier circuit. In this algorithm, every second column is taken and multiplied by 0 or +1 or +2 or -1 or -2 instead of multiplying with 0 or 1 after shifting and adding of every column of the booth multiplier. Thus, half of the partial product can be reduced using this booth algorithm. Based on the multiplier bits, the process of encoding the multiplicand is performed by radix-4 booth encoder. The overlapping is used for comparing three bits at a time. This grouping is started from least significant bit (LSB), in which only two bits of the booth multiplier are used by the first block and a zero is assumed as third bit as shown in the FIG



Booth recoding table for radix-4

Multiplier Bits Block			Recoded 1-bit pair		2 bit booth	
i+1	i	i-1	i+1	i	Multiplier Value	Partial Product
0	0	0	0	0	0	Mx0
0	0	1	0	1	1	Mx1
0	1	0	1	-1	1	Mx1
0	1	0	1	0	2	Mx2
1	0	0	-1	0	-2	Mx-2
1	0	1	-1	1	-1	Mx-1
1	1	0	0	-1	-1	Mx-1
1	1	0	0	0	0	Mx0

TABLE 1.**Booth Recoding Table for Radix-4**

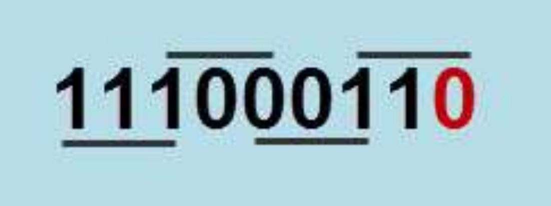


FIG 1.**Bit Pairing as per Booth Recoding**

FUSED AM IMPLEMENTATION

The main focus is on AM units which implement the operation Z=X (A+B). The conventional design of the AM operator requires that its inputs and are first driven to an adder and then the input X and the sum Y=A+B are driven to a multiplier in order to get Z. The drawback of using an adder is that it inserts a significant delay in the critical path of the AM. As there are carry signals to be propagated inside the adder, the critical path depends on the bit-width of the inputs. In order to decrease this delay, a Carry-Look-Ahead (CLA) adder can be used which, however, increases the area occupation and power dissipation. An optimized design of the AM operator is based on the fusion of the adder and the MB encoding unit into a single data path block by direct recoding of the sum Y=A+B to its MB representation. The fused Add-Multiply (FAM) component contains only one adder at the end (final adder of the parallel multiplier). As a result, significant area savings are observed and the critical path delay of the recoding process is reduced and decoupled from the bit-width of its inputs. In this work, we present a new technique for direct recoding of two numbers in the MB representation of their sum.

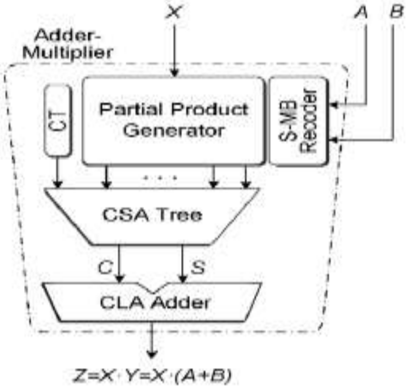


FIG 2. Fused AM scheme

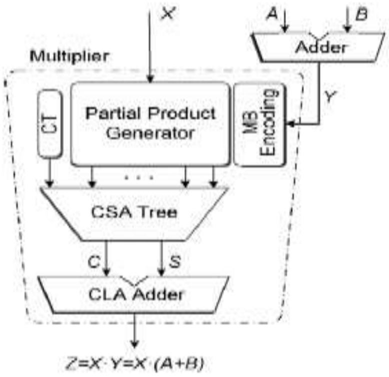


FIG 3. Conventional scheme

Consider multiplication of 2's compliment numbers X and Y with each number consisting of n=2k bits for even number of digits and n=2k+1 for odd number of digits. The multiplicand Y can be represented in Modified Booth form as follows:

$$Y = \langle y_{n-1}y_{n-2}\dots y_1y_0\rangle_{2's} = -y_{2k-1}\cdot 2^{2k-1} + \sum_{i=0}^{2k-2} y_i \cdot 2^i$$

$$= \langle \mathbf{y}_{k-1}^{MB}\mathbf{y}_{k-2}^{MB}\dots\mathbf{y}_1^{MB}\mathbf{y}_0^{MB}\rangle_{MB} = \sum_{j=0}^{k-1}\mathbf{y}_j^{MB}\cdot 2^{2j}$$

Where, $\mathbf{y}_j^{MB} = -2y_{2j+1} + y_{2j} + y_{2j-1}.$

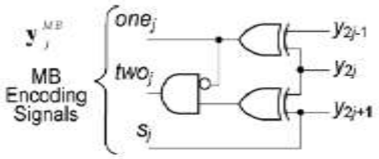
$$y_{-1} = 0$$

$$\mathbf{y}_j^{MB} \in \{-2, -1, 0, +1, +2\}, 0 \le j \le k-1$$

Each digit is represented by three bits named ‘s’, ‘one’ and ‘two’. The sign bit ‘s’ shows whether the digit is negative or positive. Signal ‘one’ shows if the absolute value of the digit is equal to 1 or not. Signal ‘two’ shows if the absolute value of the digit is equal to 2 or not. Using these three digits, we can calculate as follows,

$$\mathbf{y}_j^{MB} = (-1)^{s_j}\cdot[one_j + 2\cdot two_j].$$

$$one_j = y_{2j-1} \oplus y_{2j}$$
$$two_j = (y_{2j+1} \oplus y_{2j})\cdot\overline{one_j}$$
$$s_j = y_{2j+1}$$



(a) Boolean equations FIG 4. Gate level schematic

If we take the partial product as -2y, -y, 0, y, 2y then, we have to modify the general partial product generator. Now, every partial product point consists of two inputs (consecutive bits) from multiplicand and, based on the requirement, the output will be generated and its complements also generated in case if required.

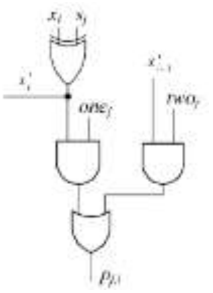


FIG 5.Generation of i-th bit of the partial product PP$_j$

The generation of i-th bit for the partial product generation is based on the logical expression:

$$p_{j,i} = ((x_i \oplus s_j) \wedge one_j) \vee ((x_{i-1} \oplus s_j) \wedge two_j)$$

Generation of the partial product is given by the equation:

$$PP_j = X \cdot \mathbf{y}_j^{MB} = \overline{p}_{j,n} 2^n + \sum_{i=0}^{n-1} p_{j,i} \cdot 2^i$$

After the partial products are generated, they are added, properly weighted, through a Wallace carry-save adder tree along with the correction term which is given by the equation:

$$Z = X \cdot Y = CT + \sum_{j=0}^{k-1} PP_j \cdot 2^{2j}$$

Correction term is found out by equation:

$$\begin{aligned} CT &= CT(low) + CT(high) = \\ &= \sum_{j=0}^{k-1} c_{\mathrm{in},j} \cdot 2^{2j} + 2^n \left(1 + \sum_{j=0}^{k-1} 2^{2j+1}\right) \end{aligned}$$

Where, $c_{\mathrm{in},j} = (one_j \vee two_j) \wedge s_j$

Finally, the carry-save output of the Wallace CSA tree is leaded to a fast Carry Look Ahead (CLA) adder to form the final result as Z=X*Y.

SUM TO MODIFIED BOOTH RECODING TECHNIQUE

In *S-MB* recoding technique, we recode the sum of two consecutive bits of the input A with two consecutive bits of B to form one MB digit. As observed, three bits are included in forming a MB digit. The most significant of them is negatively weighted while the two least significant of them have positive weight. Consequently, in order to transform the two aforementioned pairs of bits in MB form we need to use signed-bit arithmetic. For this purpose, we develop a set of bit-level signed Half Adders (HA) and Full Adders (FA) considering their inputs and outputs to be signed. In sum to modified booth recoding technique, we use signed half adders (HA* and HA**) and signed full adders (FA* and FA**)

HA** $c = \overline{p} \wedge q$, $s = p \oplus q$

FA** $c_o = ((p \vee q) \wedge \overline{c_i}) \vee (p \wedge q)$, $s = p \oplus q \oplus c_i$

HA* $c = p \vee q$, $s = p \oplus q$

FA* $c_o = ((p \vee \overline{q}) \wedge c_i) \vee (p \wedge \overline{q})$, $s = p \oplus q \oplus c_i$

S-MB1 recoding scheme for even numbers of bits and odd number of bits are done using the following diagram:

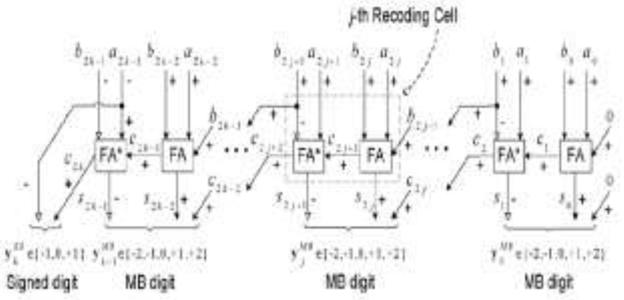


FIG 6.

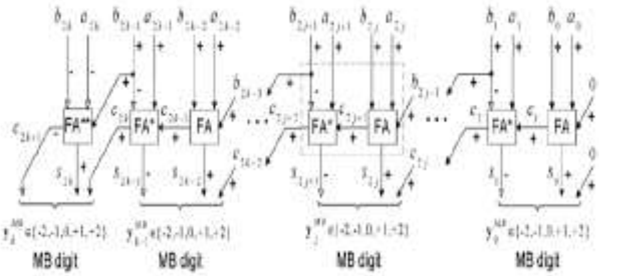


FIG 7.

S-MB2 recoding scheme for even numbers of bits and odd number of bits are done using the following diagram:

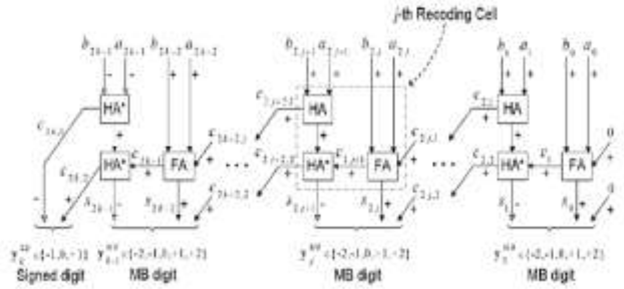


FIG 8.

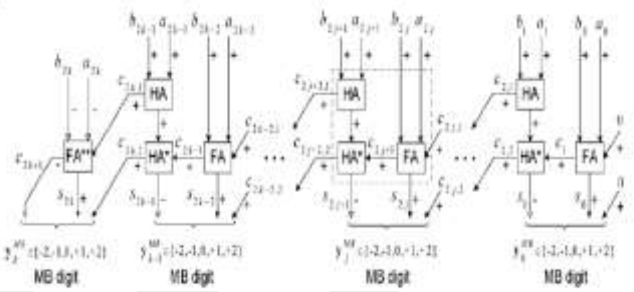


FIG 9.

S-MB3 recoding scheme for even numbers of bits and odd number of bits are done using the following diagram:

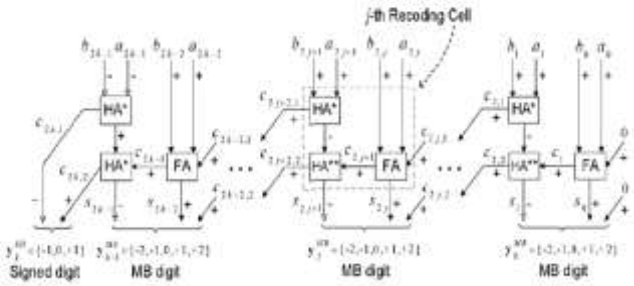


FIG 10.

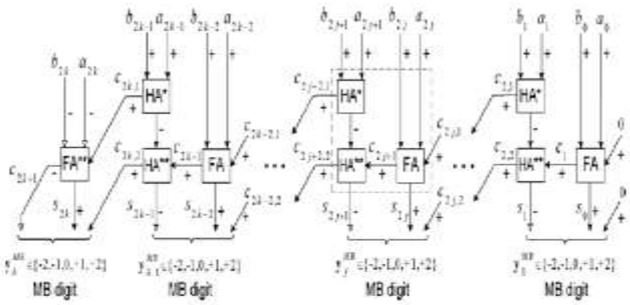


FIG 11.

4:2 COMPRESSORS

A 4-2 compressor consists of five inputs and three outputs. It is called compressor, since it compress four partial products into two. This can be implemented with two stages of full adders (FA) connected in series. By using compressors in the circuit, produced considerable reduction in area as well as delay.

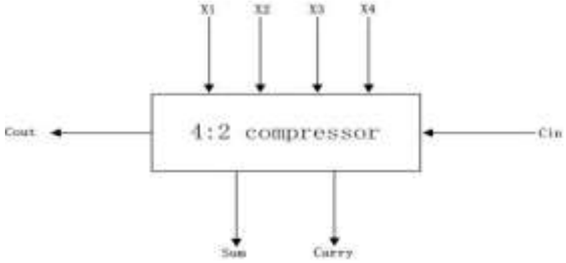


FIG 12.

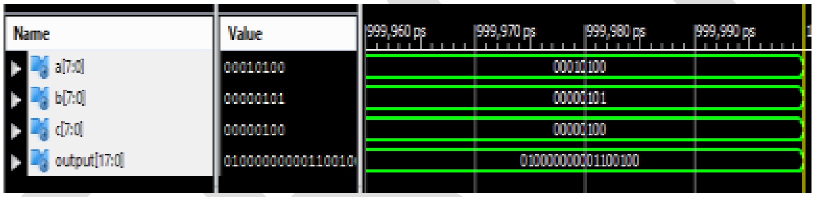
SIMULATION RESULTS

Z = X (A + B)

Here, N = 8 (Even no: of bits)

A = 9 (“00001001”)   ,   B = 11 (“00001011”)   ,   X = 5 (“00000101”)

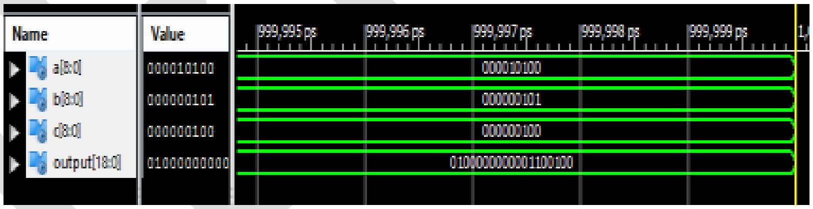
Z = 5 (9+11) = 5 (20) = 100 (“01100100”)



Name	Value
a[7:0]	00010100
b[7:0]	00000101
c[7:0]	00000100
output[17:0]	01000000000110010

No: of slices – 4% (86 out of 1920)   Delay – 25.253 ns

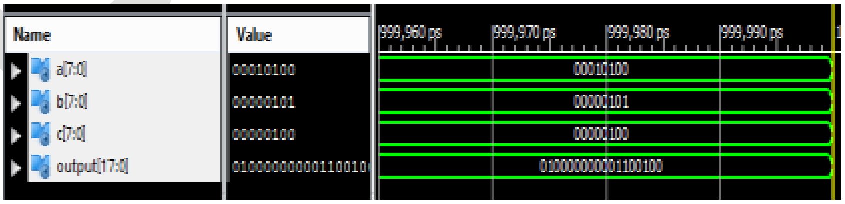
FIG 13. Simulation results for modified SMB1 even no: of bits



Name	Value
a[8:0]	000010100
b[8:0]	000000101
c[8:0]	000000100
output[18:0]	01000000000

No: of slices – 5% (110 out of 1920)   Delay – 31.822 ns

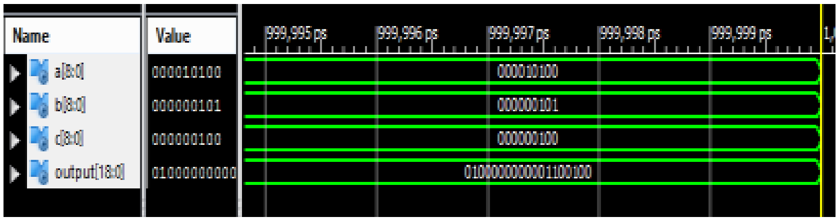
FIG 14. Simulation results for modified SMB1 odd no: of bits



Name	Value
a[7:0]	00010100
b[7:0]	00000101
c[7:0]	00000100
output[17:0]	01000000000110010

No: of slices – 4% (85 out of 1920)   Delay – 25.331 ns

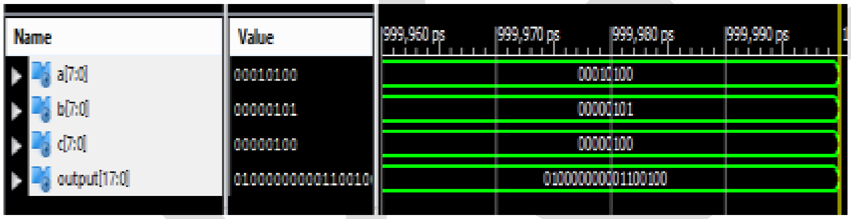
FIG 15. Simulation results for modified SMB2 even no: of bits



Name	Value
a[8:0]	000010100
b[8:0]	000000101
c[8:0]	000000100
output[18:0]	01000000000

No: of slices – 5% (110 out of 1920) Delay – 32.064 ns

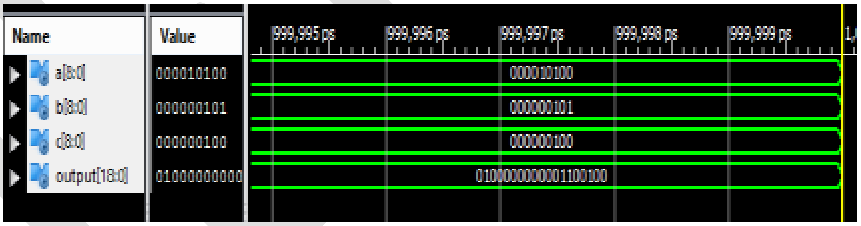
FIG 16. Simulation results for modified SMB2 odd no: of bits



Name	Value
a[7:0]	00010100
b[7:0]	00000101
c[7:0]	00000100
output[17:0]	010000000000110010

No: of slices – 4% (82 out of 1920) Delay – 25.318 ns

FIG 17. Simulation results for modified SMB3 even no: of bits



Name	Value
a[8:0]	000010100
b[8:0]	000000101
c[8:0]	000000100
output[18:0]	01000000000

No: of slices – 5% (107 out of 1920) Delay – 31.921 ns

FIG 26. Simulation results for modified SMB3 odd no: of bits

CONCLUSION

The modified S-MB recoding schemes focuses on optimizing the design of the Fused-Add Multiply (FAM) operator. Also, proposing a structured technique for the direct recoding of the sum of two numbers to its MB form. Explore alternative designs of the proposed *S-MB* recoder and compare them to the existing ones. The proposed recoding schemes, when they are incorporated in FAM designs, yield considerable performance improvements in comparison with the most efficient recoding schemes found in literature. Sum to modified booth recoding scheme (S-MB1) for optimizing fused add-multiply unit has been presented. Proposed scheme allows considerable improvement in performance.S-MB1 scheme, when incorporated with FAM designs decreases critical path delay as well as area consumption compared to conventional recoding schemes used.

REFERENCES:

[1] C. N. Lyu and D. W. Matula, “Redundant binary Booth recoding,” in*Proc. 12th Symp. Comput. Arithmetic*, 1995, pp. 50–57.

[2] W.C. Yeh, “Arithmetic Module Design and its Application to FFT,” Ph.D. dissertation, Dept. Electron. Eng., National Chiao-Tung University, Chiao-Tung, 2001.

[3] M. Daumas and D. W. Matula, “A Booth multiplier accepting both a redundant or a non redundant input with no additional delay,” in *Proc. IEEE Int. Conf. on Application-Specific Syst., Architectures, and Processors*, 2000, pp. 205–214.

[4] R. Zimmermann and D. Q. Tran, “Optimized synthesis of sum-of-products,” in *Proc. Asilomar Conf. Signals, Syst. Comput.*, Pacific Grove, Washington, DC, 2003, pp. 867–872.

[5] K. Tsoumanis, S. Xydis, C. Efstathiou, N. moschopoulus and K. Pekmestzi, “ An Optimized Modified Booth Recoder for Efficient Design of the Add-Multiply Unit” , in *Archemidis III: Funding of Research Groupsin TEI*, Athens, 2014